Parallel multigrid methods for parabolic partial differential equations and applications

Feng Wei Yang

Department of Mathematics
University of Sussex

The Leverhulme Trust

F.W.Yang@sussex.ac.uk

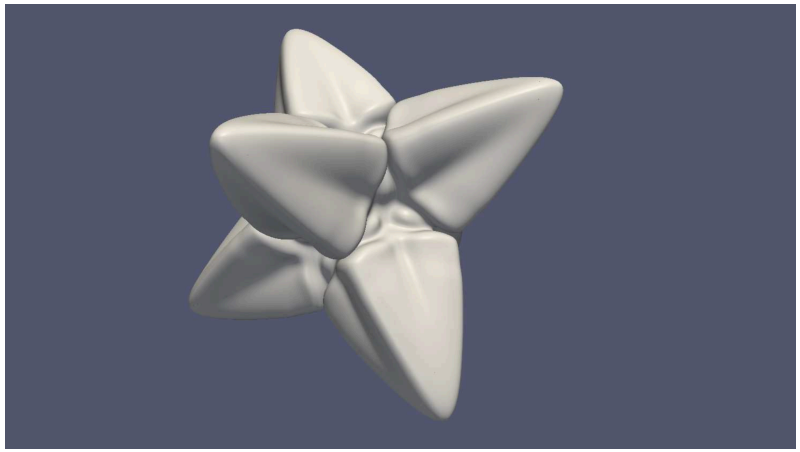
1 October 2015

Objectives

To solve complex non-linear parabolic systems by applying:

- Cartesian Grids (FDM)
- Implicit Schemes (BDFs)
- Nonlinear Multigrid Method with Full Approximation Scheme (FAS)
- Adaptive Mesh Refinement (AMR)
- Adaptive Time-Stepping (ATS)
- Parallel Technique

An example of the so-called "stiff" problems



Binary alloy solidification in 3D

P. Bollada, C.E. Goodyer, P.K. Jimack, A.M. Mullis, F.W. Yang
Journal of Computational Physics, 2015

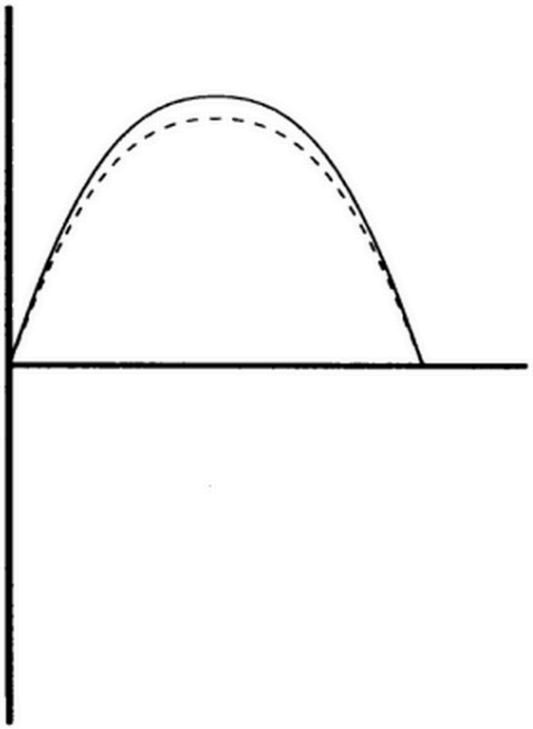
Outline

- Multigrid methods
- Thin film model from Gaskell et al.
- Optimal control model with geometric evolution laws for whole cell tracking

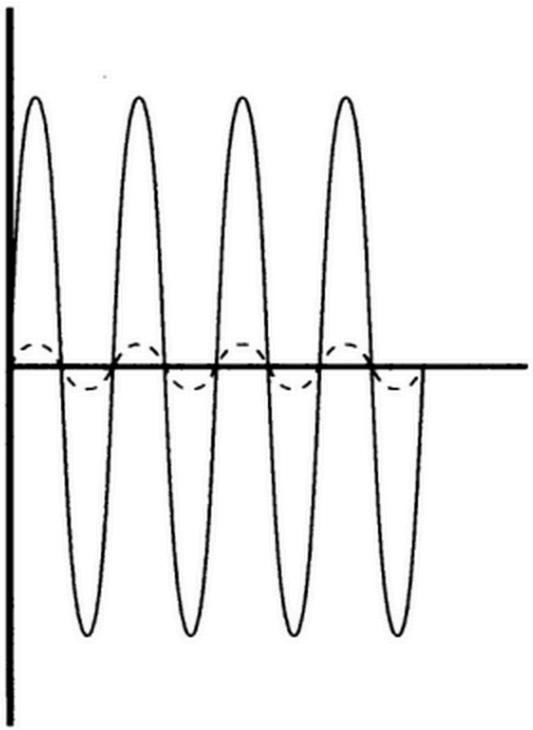
Jacobi/Gauss-Seidel iterative methods

- Well-known methods
- Require diagonally-dominant matrices
- Typically have complexity of $\mathcal{O}(n^2)$ for general sparse matrices
- ...
- Smoothing property

Low frequency of error

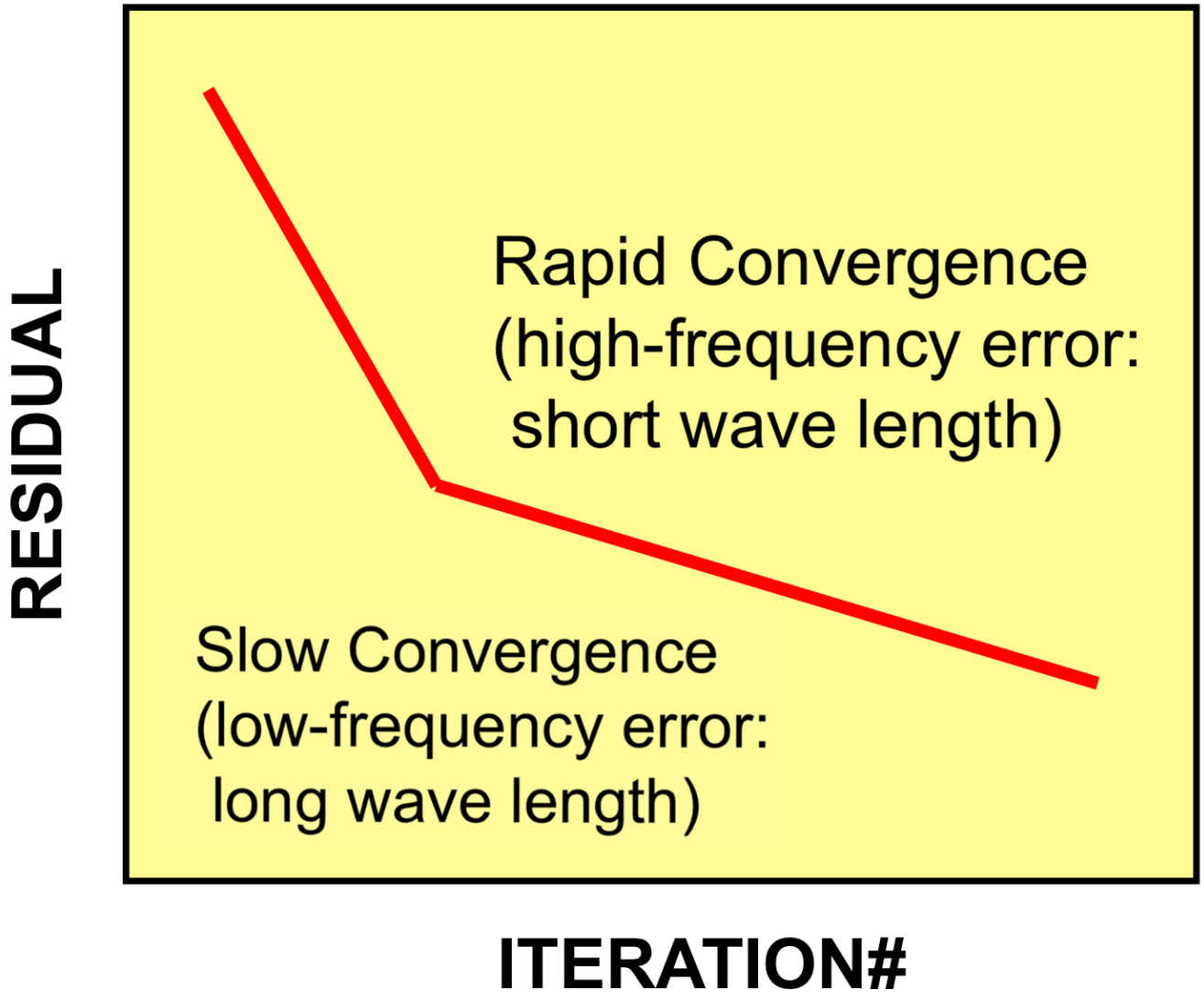


High frequency of error



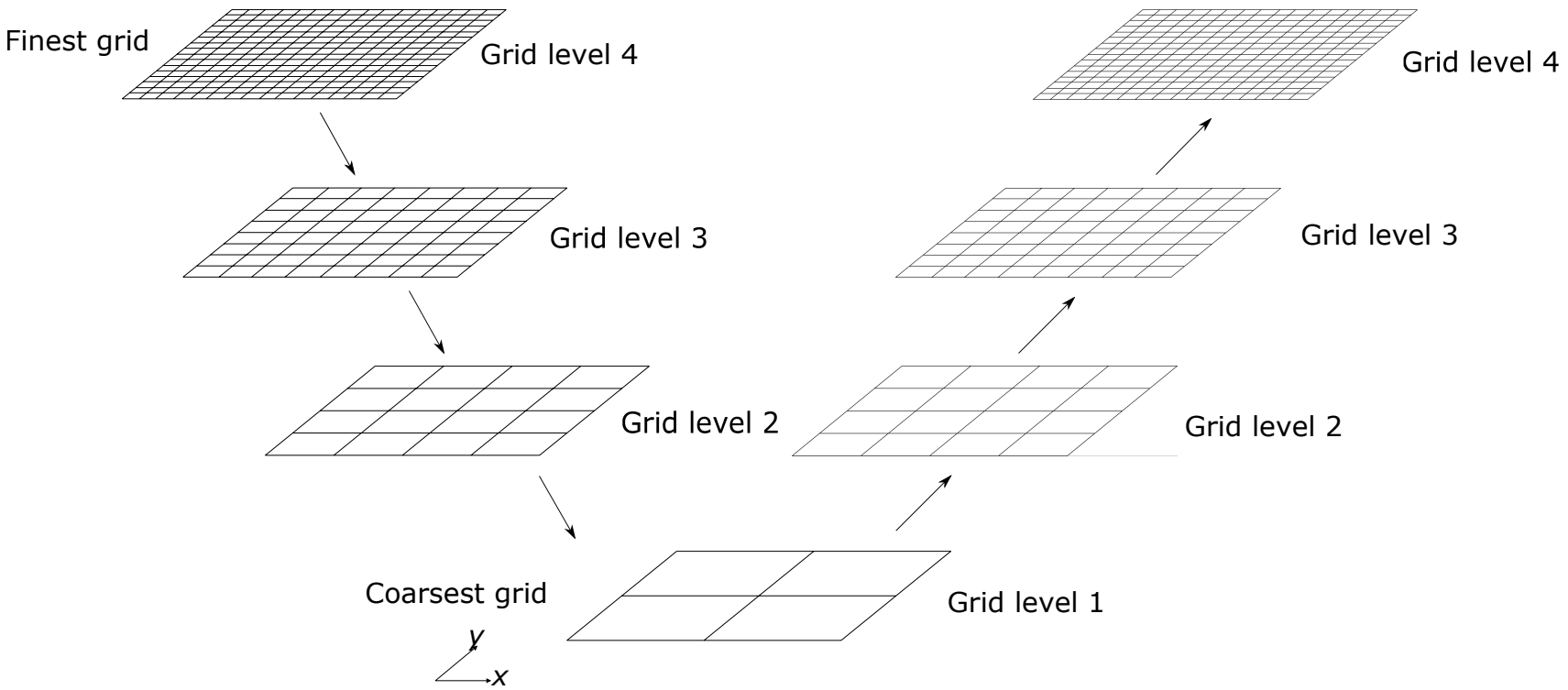
S.H. Lui *Numerical Analysis of Partial Differential Equations*, 2011

Convergence of a typical Jacobi iterative method



source: nkl.cc.u-tokyo.ac.jp

Multigrid V-cycle



Linear multigrid

A linear problem:

$$Au = b, \tag{1}$$

exact error can be obtained as

$$E = u - v, \tag{2}$$

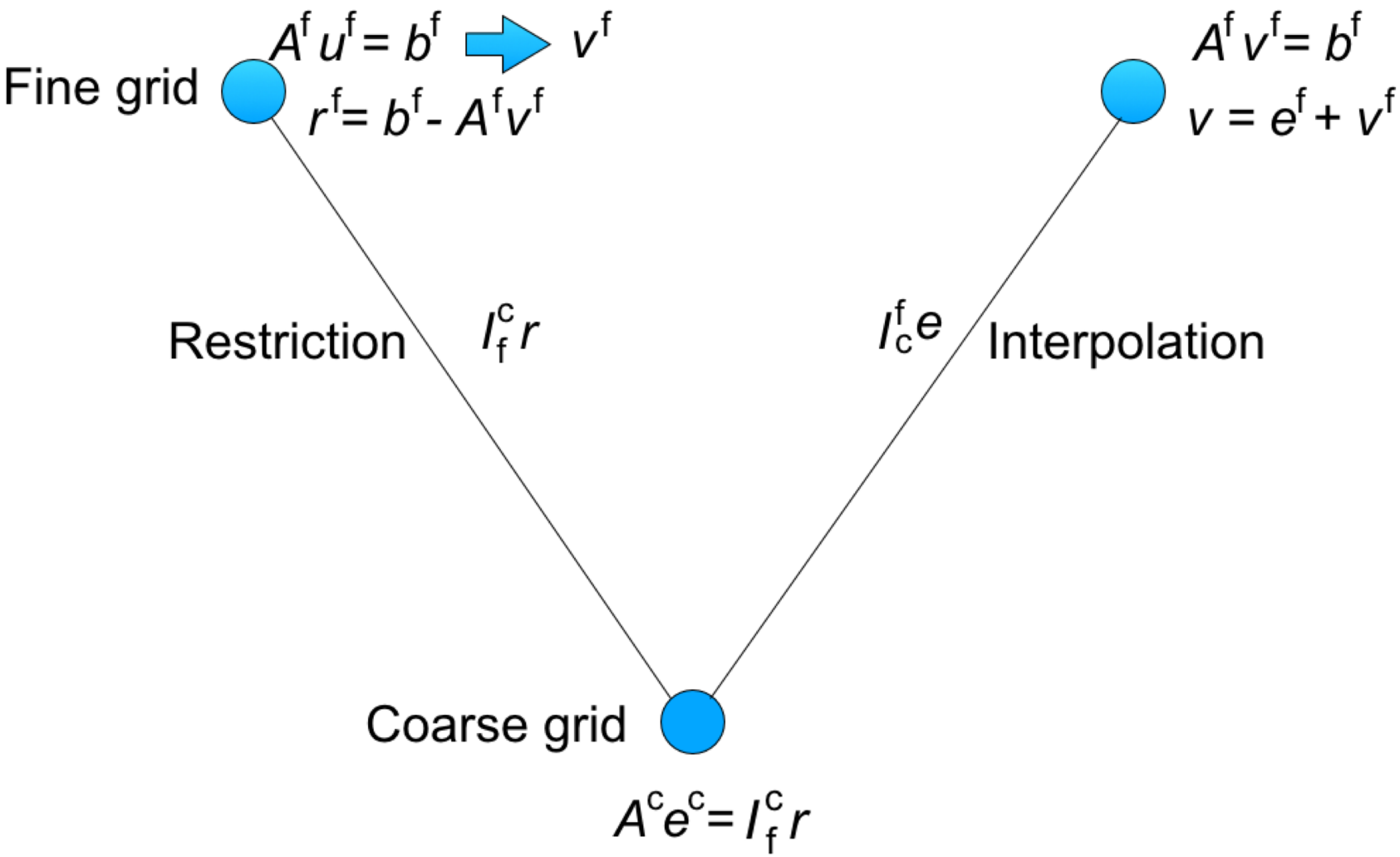
residual can be calculated as:

$$r = b - Av. \tag{3}$$

Error equation:

$$\begin{aligned} AE &= A(u - v) \\ &= Au - Av \\ &= b - Av \\ &= r. \end{aligned} \tag{4}$$

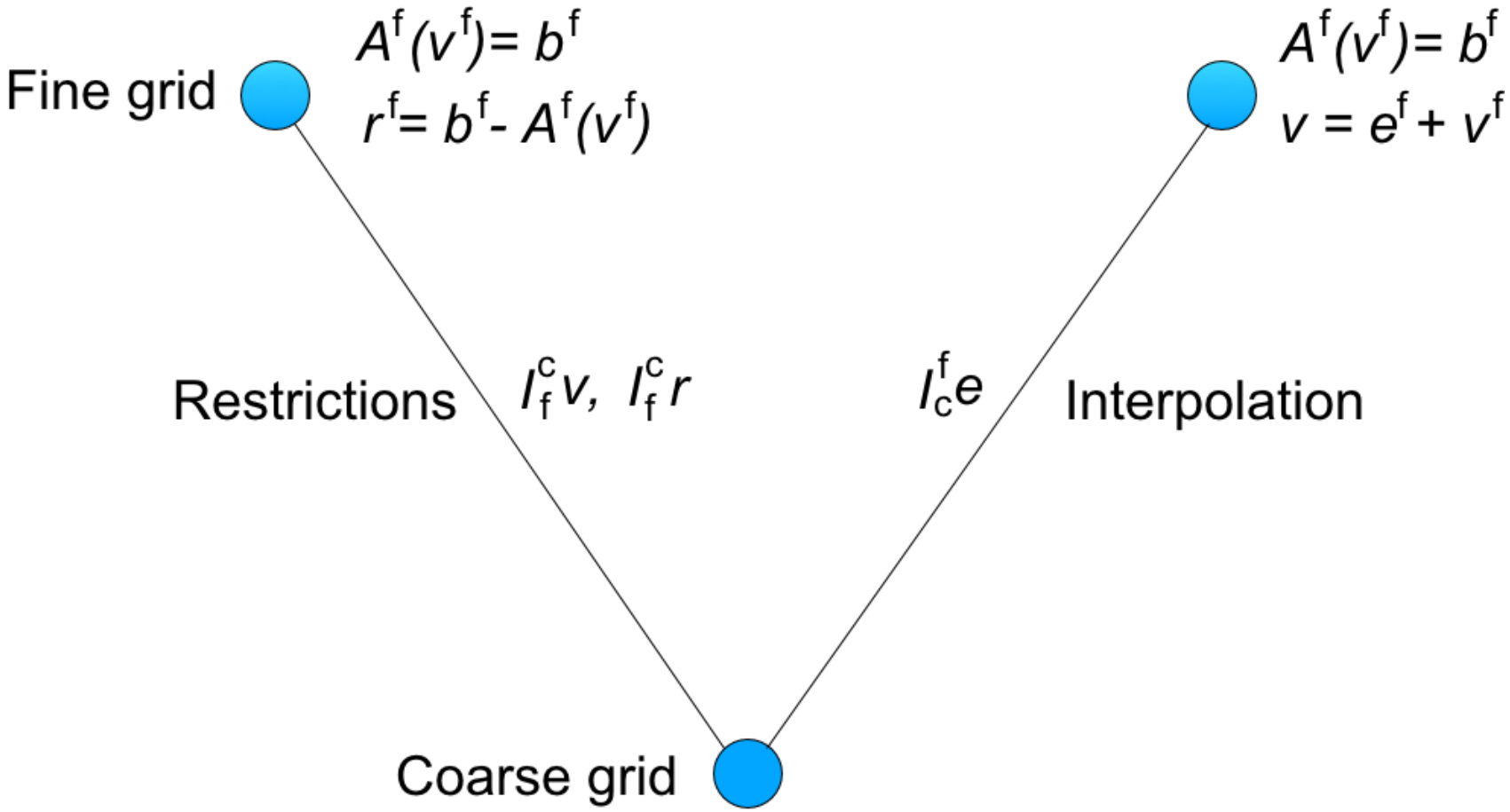
Linear multigrid



Nonlinear multigrid

- The Error Equation (4) does not exist in a nonlinear case
- Full Approximate Scheme (FAS)
- For problem on coarser grids, a modified RHS is included

Nonlinear multigrid



$$A^c(v^c) = f^c + (I_f^c r - (f^c - A^c(I_f^c v)) = I_f^c r + I_f^c v$$

$$e^c = v^c - I_f^c v$$

A nonlinear point-wise smoother

Let's consider our nonlinear problem:

$$A(v) = f.$$

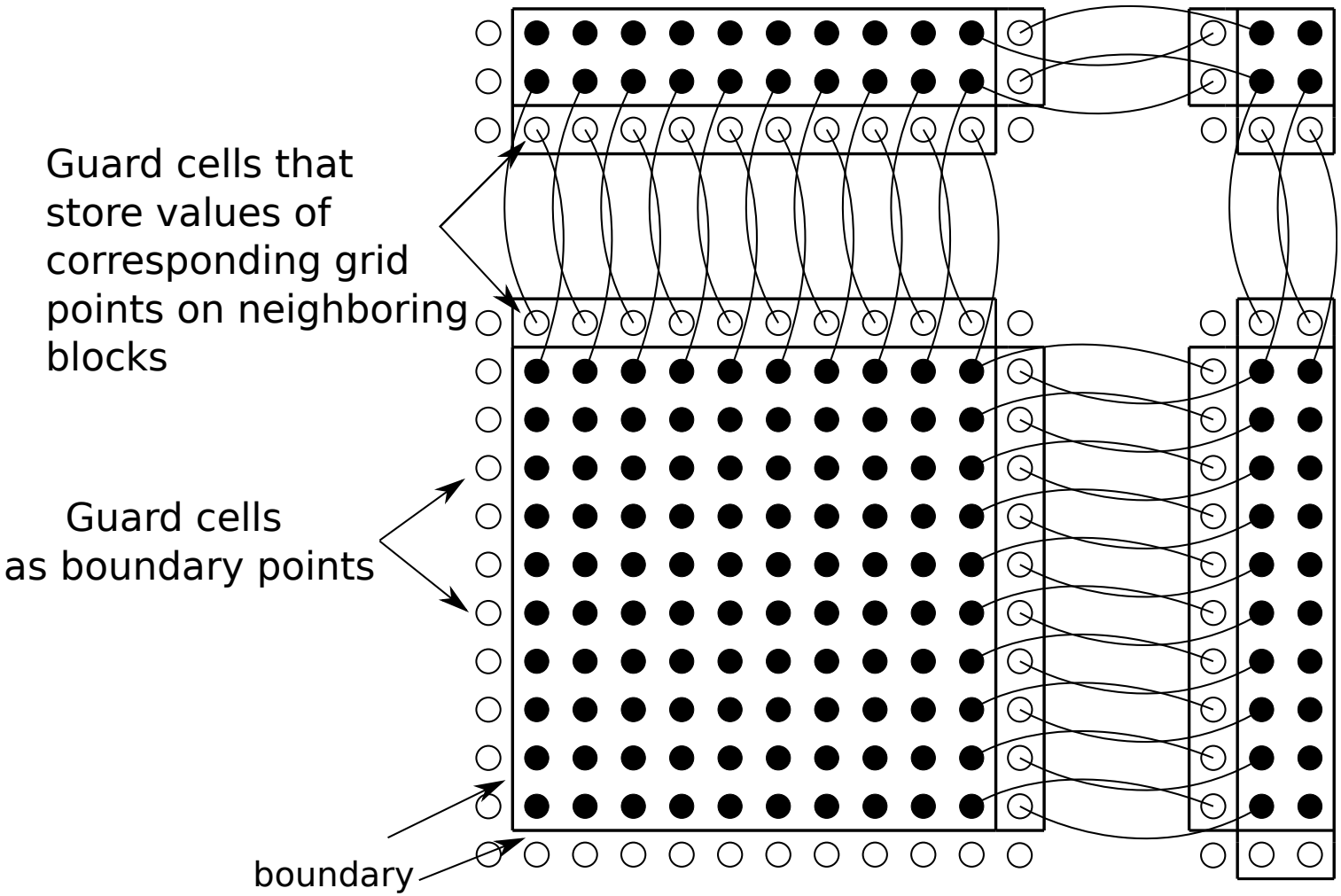
It can be rewritten as:

$$\mathcal{F}(v) = 0.$$

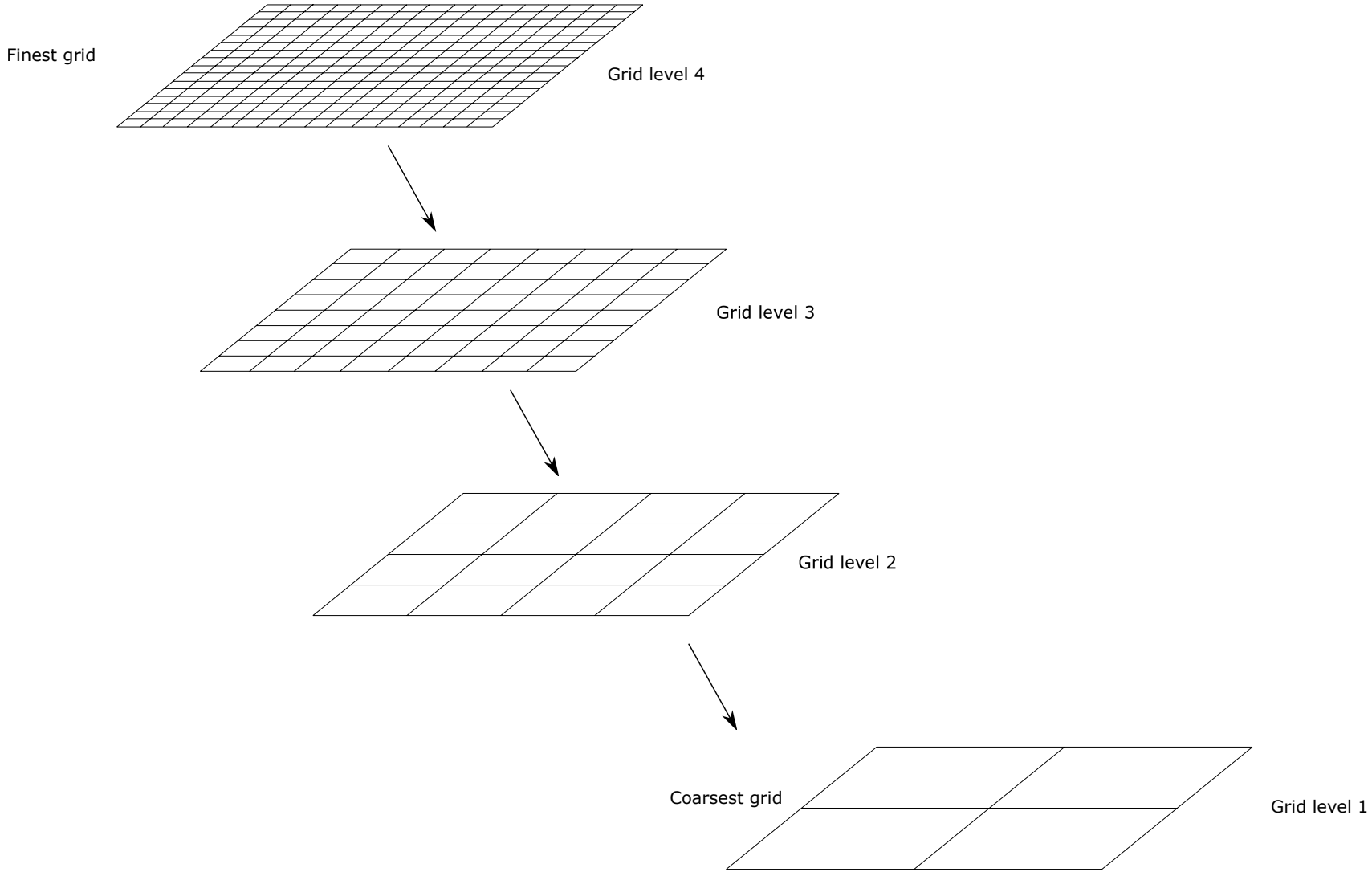
Then the Newton-like nonlinear point-wise smoother at a particular grid point $(i, j) \in \Omega$ can be the following:

$$v_{i,j}^{\ell+1,t+1} = v_{i,j}^{\ell,t+1} - \frac{\mathcal{F}(v)}{\mathcal{F}'(v_{i,j}^{\ell,t+1})}.$$

Domain decomposition and guard cells



Multigrid in parallel

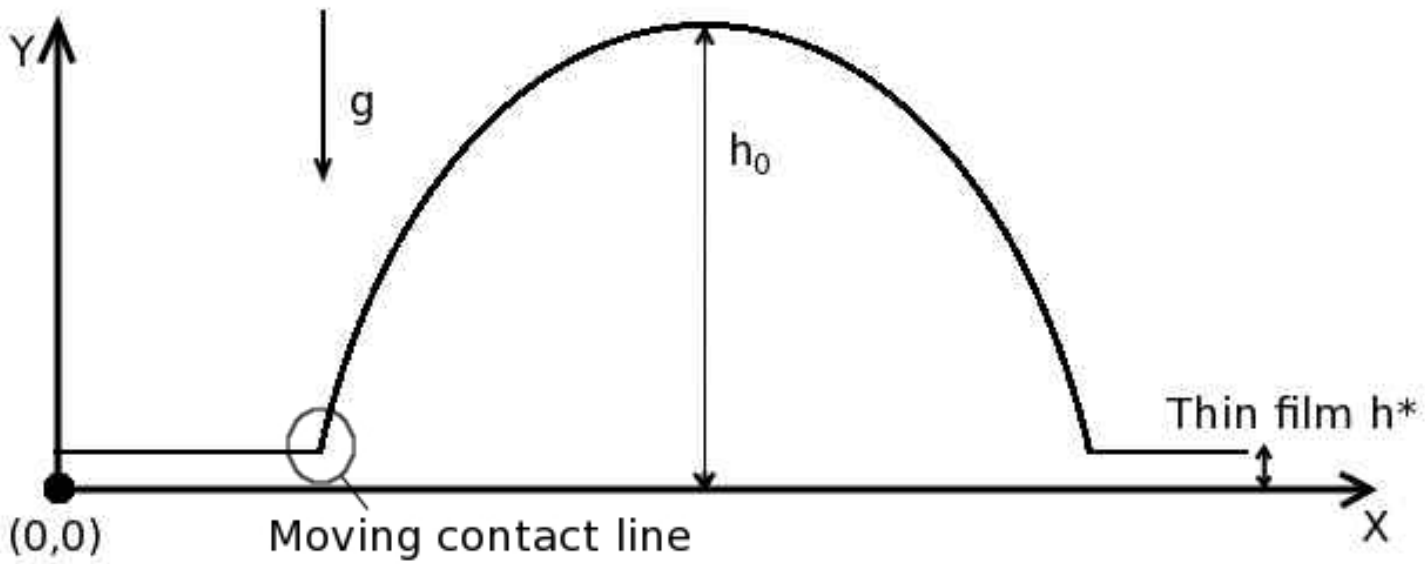


Droplet spreading model

$$\frac{\partial h}{\partial t} = \frac{\partial}{\partial x}\left[\frac{h^3}{3}\left(\frac{\partial p}{\partial x} - \frac{B_o}{\epsilon} sin\alpha\right)\right] + \frac{\partial}{\partial y}\left[\frac{h^3}{3}\left(\frac{\partial p}{\partial y}\right)\right]$$
$$p = -\triangle(h) - \Pi(h) + B_o h \cos\alpha$$

with boundary conditions:

$$h = h^* \quad \partial_n p = 0 \quad \text{on} \quad \partial\Omega$$



Gaskell et al. *Int. J. Numer. Meth. Fluids*, 45:1161-1186, 2004

Our solver

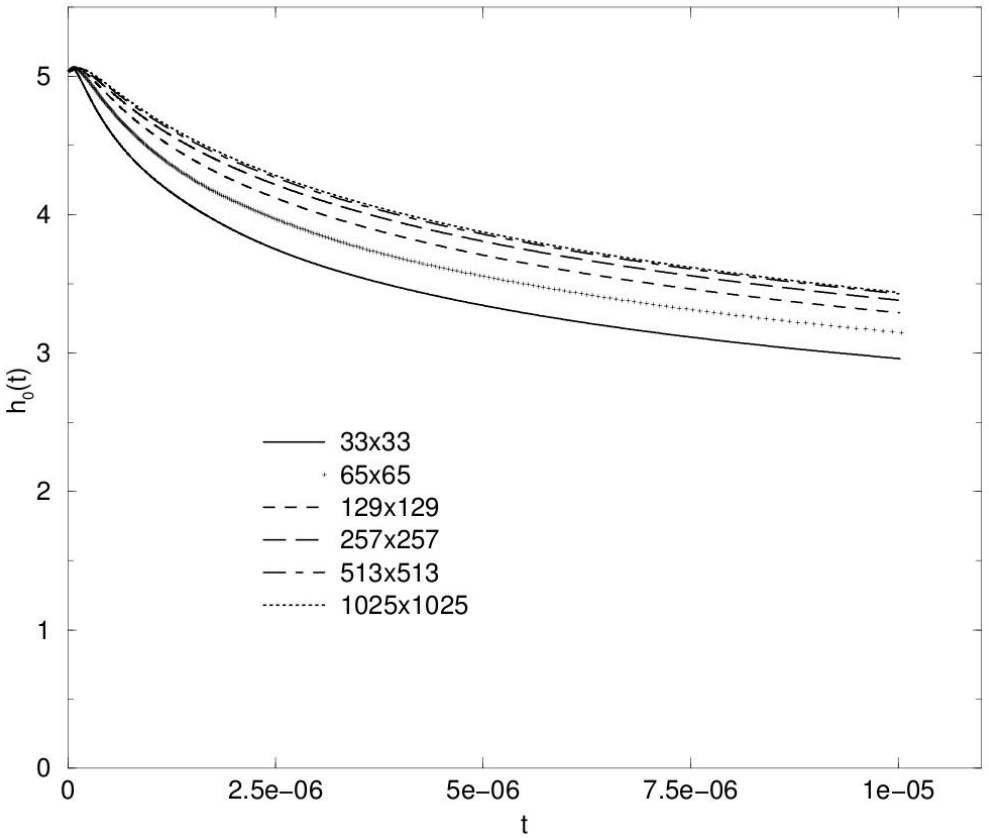
- Cell-centred 2^{nd} order finite difference method
- PARAMESH library for mesh generation and AMR
- Fully implicit BDF2 method with adaptive time-stepping
- MLAT variation of FAS multigrid at each time-step
- Newton-block 2×2 Red-Black (weighted) Gauss-Seidel smoother
- Full weighting restriction and bilinear interpolation
- Parallelisation through domain decomposition

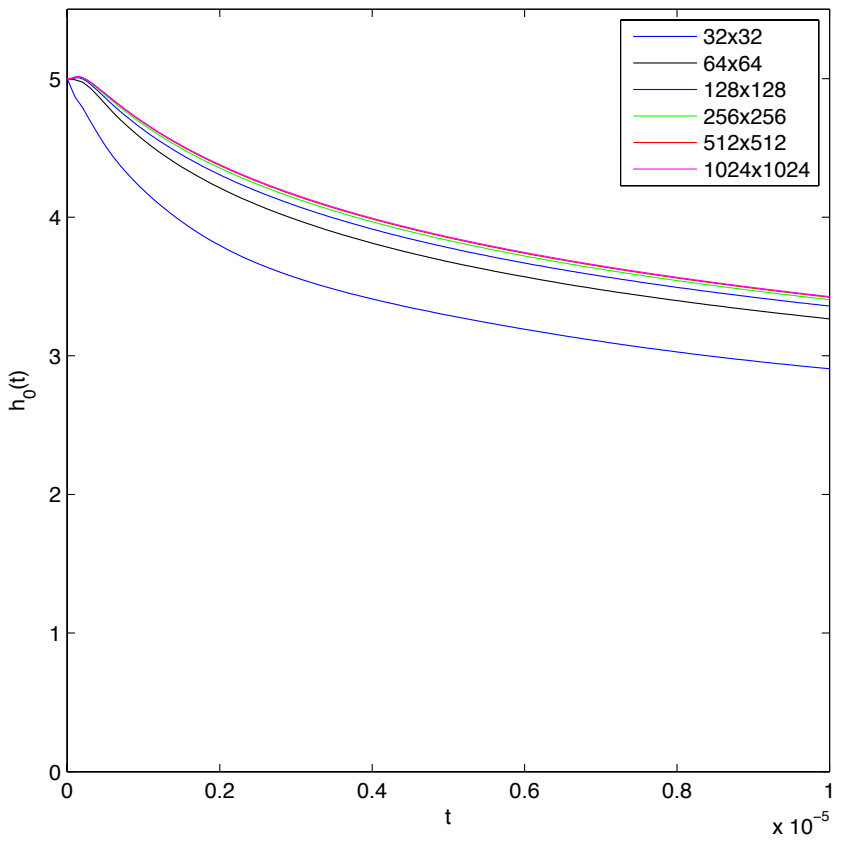
Newton-block smoother

Update at a grid point (i, j):

$$\begin{pmatrix} h^{\ell+1,t+1} \\ p^{\ell+1,t+1} \end{pmatrix}_{i,j} = \begin{pmatrix} h^{\ell,t+1} \\ p^{\ell,t+1} \end{pmatrix}_{i,j} - \begin{pmatrix} \frac{\partial \mathcal{F}_h}{\partial h_{i,j}^{t+1}} & \frac{\partial \mathcal{F}_h}{\partial p_{i,j}^{t+1}} \\ \frac{\partial \mathcal{F}_p}{\partial h_{i,j}^{t+1}} & \frac{\partial \mathcal{F}_p}{\partial p_{i,j}^{t+1}} \end{pmatrix}^{-1} \begin{pmatrix} \mathcal{F}_{h\,i,j}(h,p) \\ \mathcal{F}_{p\,i,j}(h,p) \end{pmatrix}$$

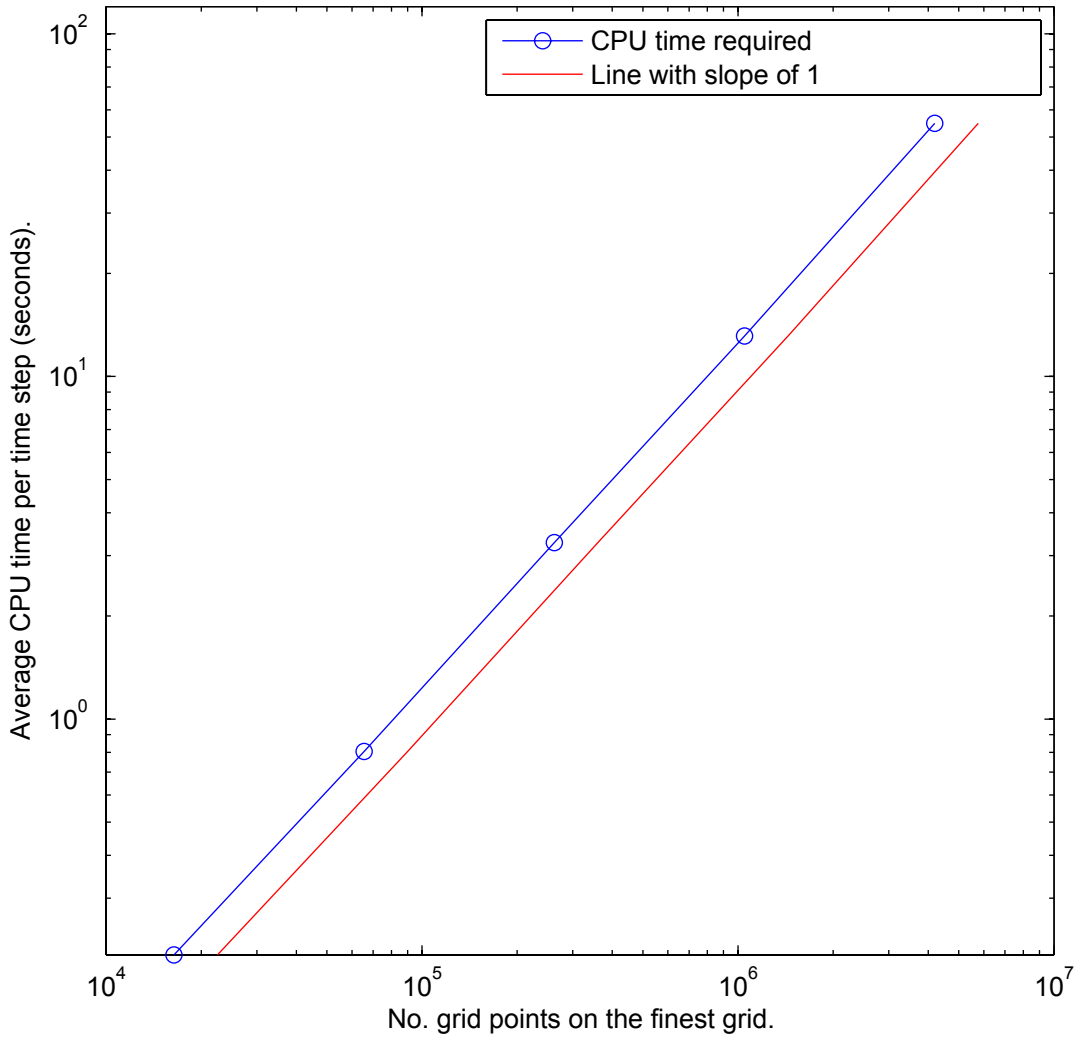
Validation





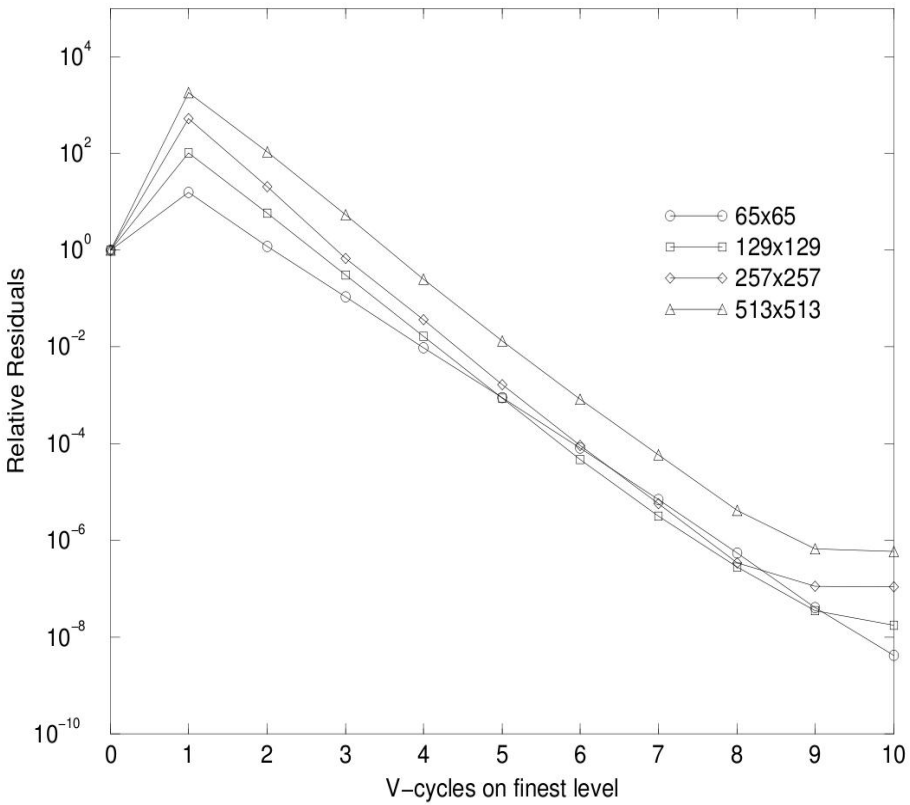
Results from Gaskell et al. on the left and our results on the right.

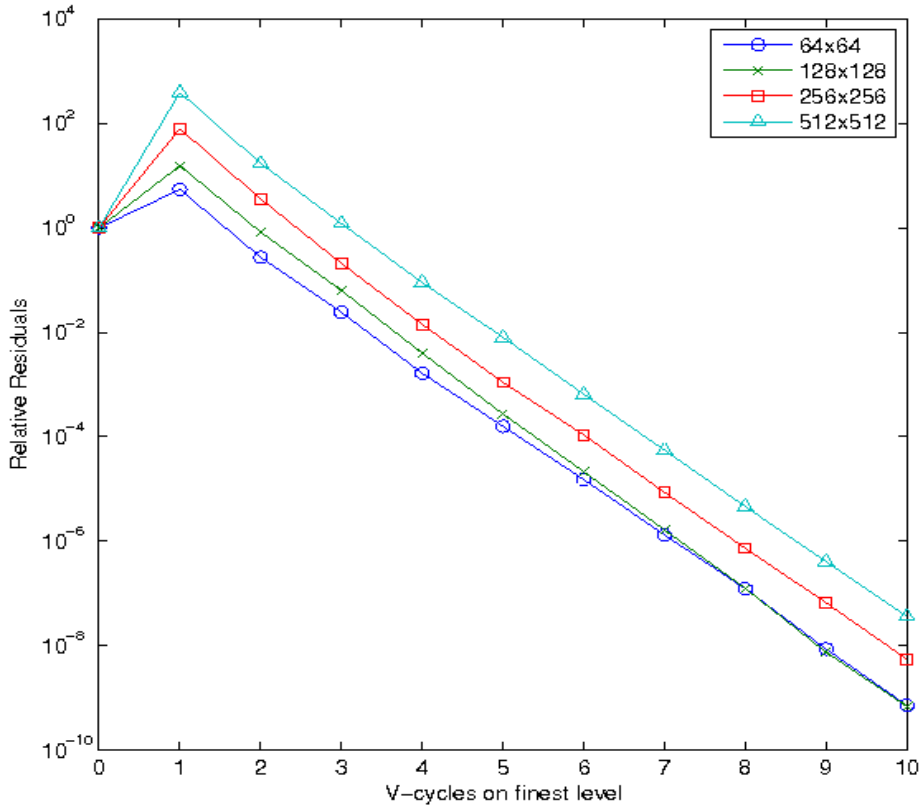
Multigrid linear complexity



A log-log plot demonstrating the linear complexity of multigrid.

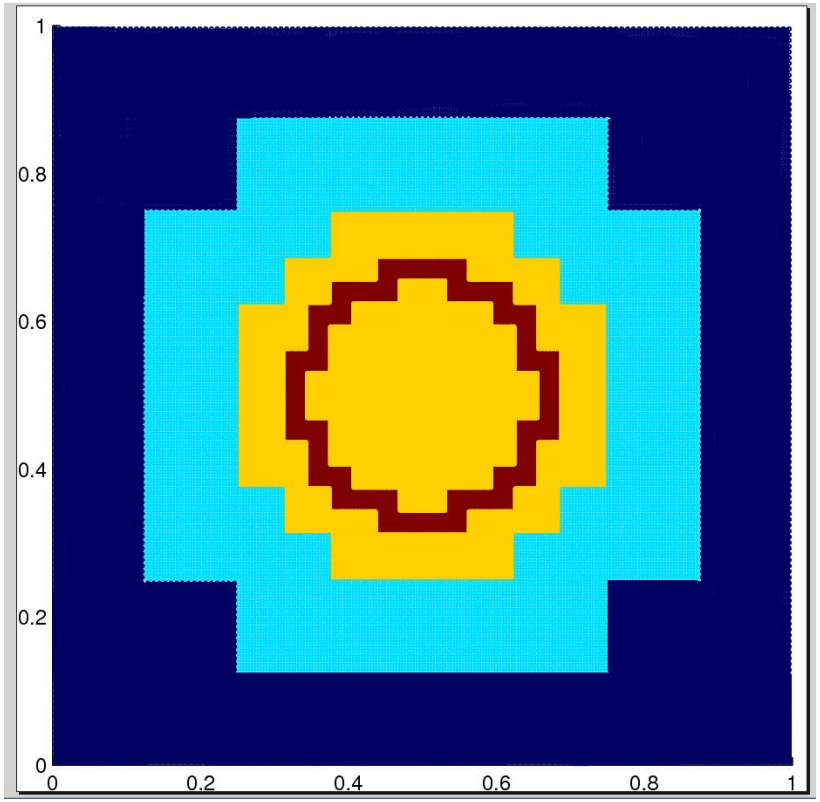
Multigrid performance

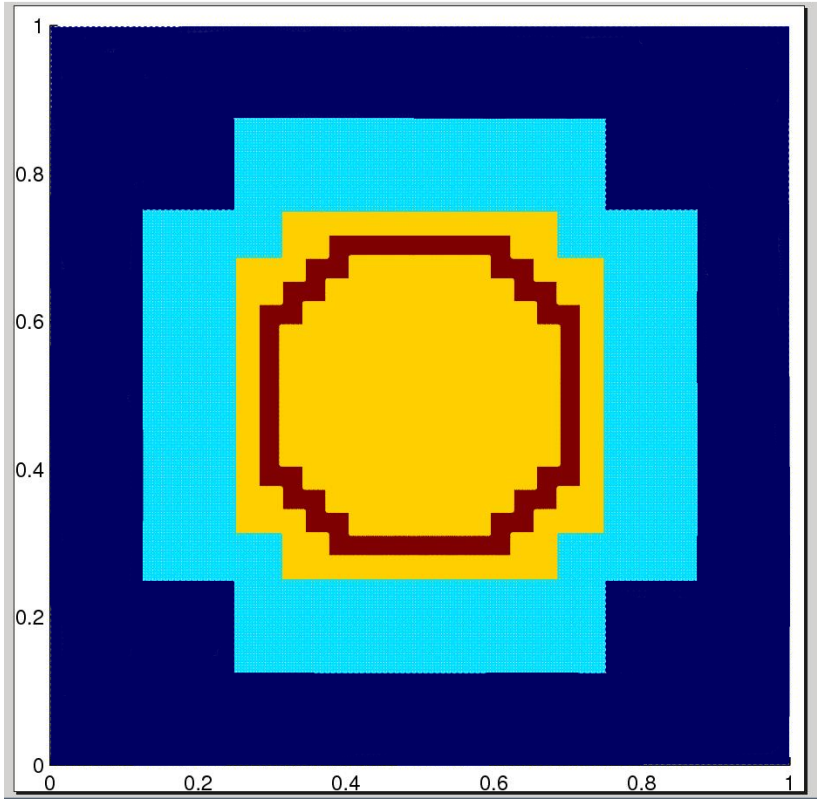




Results from Gaskell et al. on the left and our results on the right.

AMR

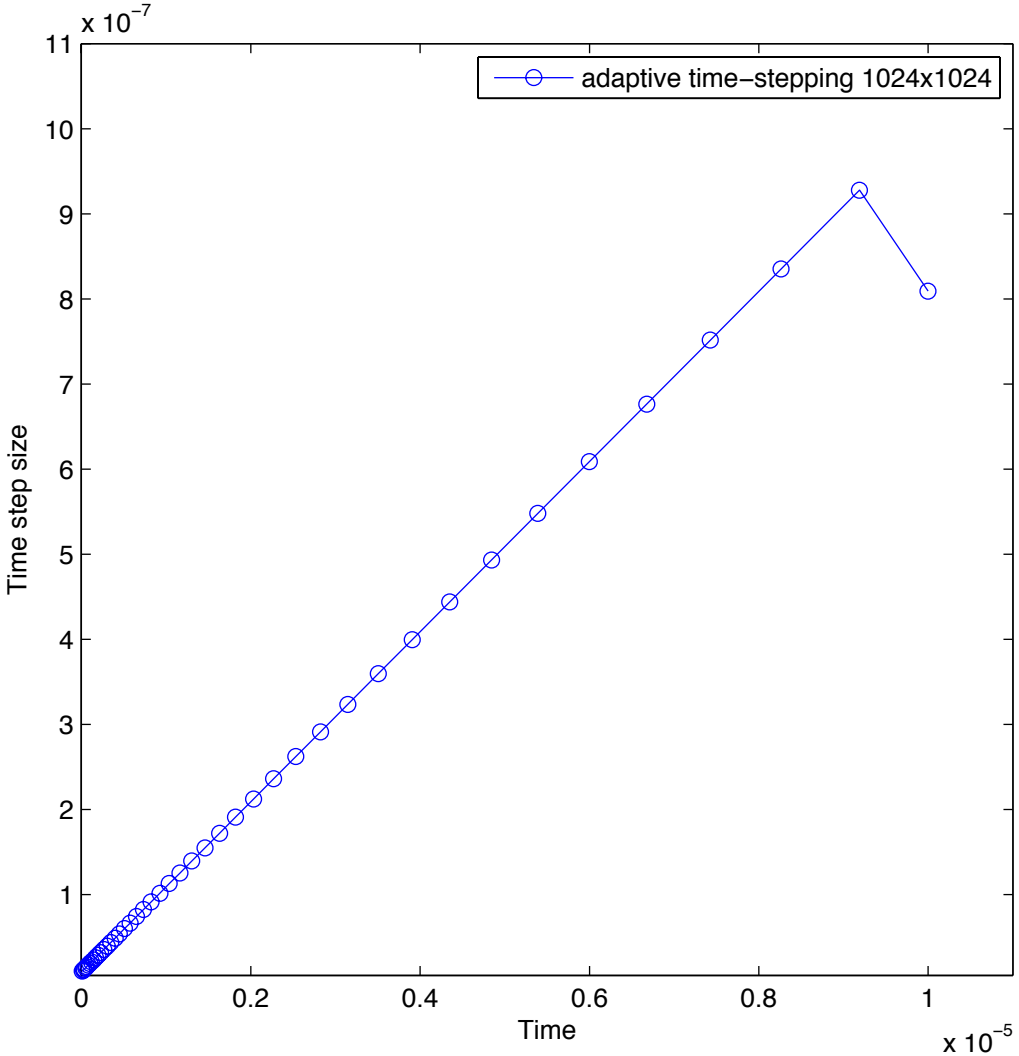




AMR with initial condition on the left and final solution on the right.

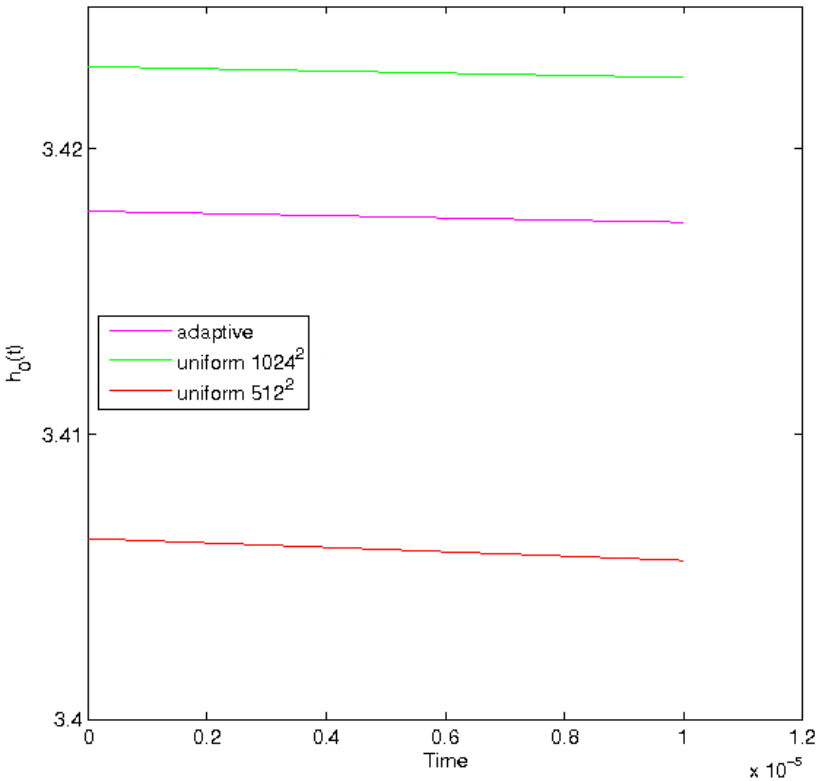
Animation

Adaptive time-stepping



Evolution of δt during $T = [0, 1 \times 10^{-5}]$.

Adaptive multigrid solver



Cases	No. leaf nodes
Uniform 1024^2	1,048,576
AMR	168,480

Cases	No. time steps	CPU time (seconds)
Fixed δt	1000	16721.3
ATS	45	574.4

F. Yang et al. *Advances in Engineering Software*, in review, 2015

Optimal control with geometric evolution laws for whole cell tracking

K.N. Blazakis, A. Madzvamuse, C. Reyes-Aldasoro, V. Styles, C. Venkataraman

"Whole cell tracking through the optimal control of geometric evolution laws"

Journal of Computational Physics, 2015

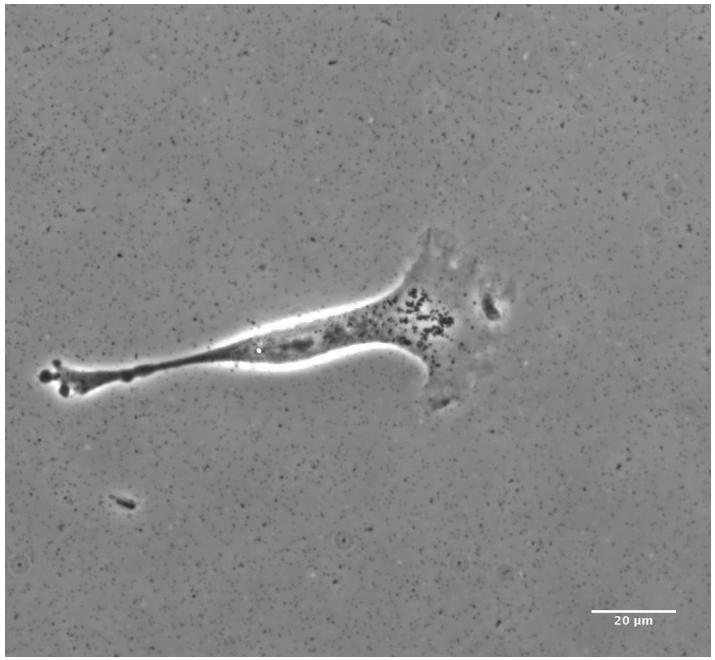
F. Yang, C. Venkataraman, V. Styles, A. Madzvamuse

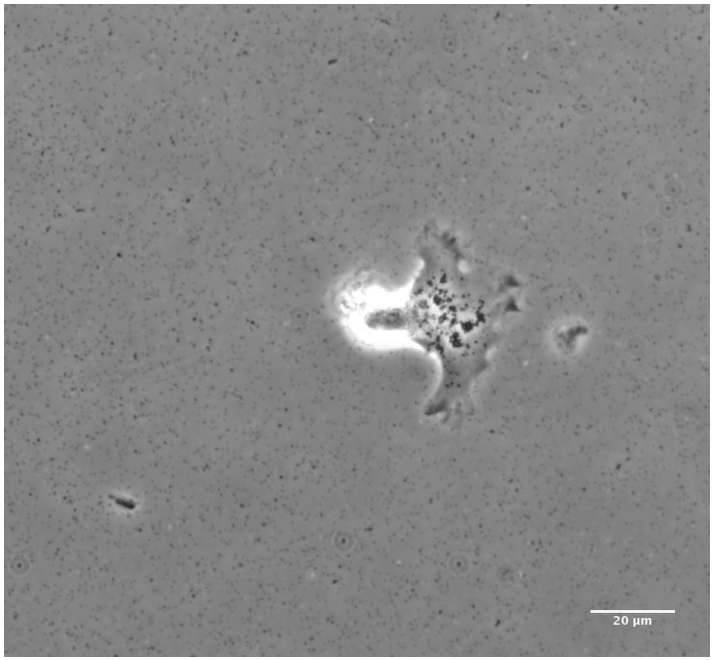
"A robust and efficient adaptive multigrid solver for the optimal control of phase field formulations of geometric evolution laws"

in review, 2015

Model objectives

To track the morphology of cells and reconstruct their movements:





V. Peschetola et al. *Cytoskeleton*, 2013

What is our signature

Particle tracking:

- The morphology of cells are not considered
- Manually tracking is slow
- Automatic tracking algorithms are often flawed
  - Segmentation is suboptimal for real data
  - Tracking through patten recognition is challenging

Pure geometric math models:

- Resolution of the data matters
- Typically no cell-setting is considered
- It is a complicated procedure to obtain the results
- Computational power and advanced numerical methods have to be included for 3-D real-life cell tracking

Our optimal control model

The volume conserved mean curvature flow:

$$\begin{cases} \boldsymbol{V}(\boldsymbol{x},t) & = (-\sigma H(\boldsymbol{x},t) + \eta(\boldsymbol{x},t) + \lambda_V(t))\,\boldsymbol{\nu}(\boldsymbol{x},t) \ \text{ on } \ \Gamma(t), t \in (0,T], \\ \Gamma(0) = \Gamma^0. \end{cases}$$

The phase-field approximation of the above equation - Allen-Cahn:

$$\begin{cases} \partial_t \phi(\boldsymbol{x},t) & = \triangle \phi(\boldsymbol{x},t) - \frac{1}{\epsilon^2} G'(\phi(\boldsymbol{x},t)) - \frac{1}{\epsilon}(\eta(\boldsymbol{x},t) - \lambda(t)) \ \text{ in } \ \Omega \times (0,T], \\ \nabla\phi \cdot \boldsymbol{\nu}_\Omega & = 0 \ \text{ on } \ \partial\Omega \times (0,T], \\ \phi(\cdot,0) & = \phi^0 \ \text{ in } \ \Omega. \end{cases}$$

Our optimal control model cont.

The objective functional:

$$J(\phi, \eta) = \frac{1}{2} \int_{\Omega} \left(\phi(\boldsymbol{x}, T) - \phi_{obs}(\boldsymbol{x})\right)^2 d\boldsymbol{x} + \frac{\theta}{2} \int_0^T \int_{\Omega} \eta(\boldsymbol{x}, t)^2 d\boldsymbol{x} dt,$$

and now we solve the minimisation problem:

$$min_{\eta} J(\phi, \eta), \quad \text{with } J \text{ given above.}$$

Our optimal control model cont.

The adjoint equation to help computing the derivative of the objective functional:

$$\begin{cases} \partial_t p(\boldsymbol{x}, t) = -\triangle p(\boldsymbol{x}, t) + \epsilon^{-2} G''\left(\phi\left(\boldsymbol{x}, t\right)\right) p(\boldsymbol{x}, t) & \text{in } \Omega \times [0, T), \\ p(\boldsymbol{x}, T) = \phi(\boldsymbol{x}, T) - \phi_{obs}(\boldsymbol{x}) & \text{in } \Omega, \end{cases}$$

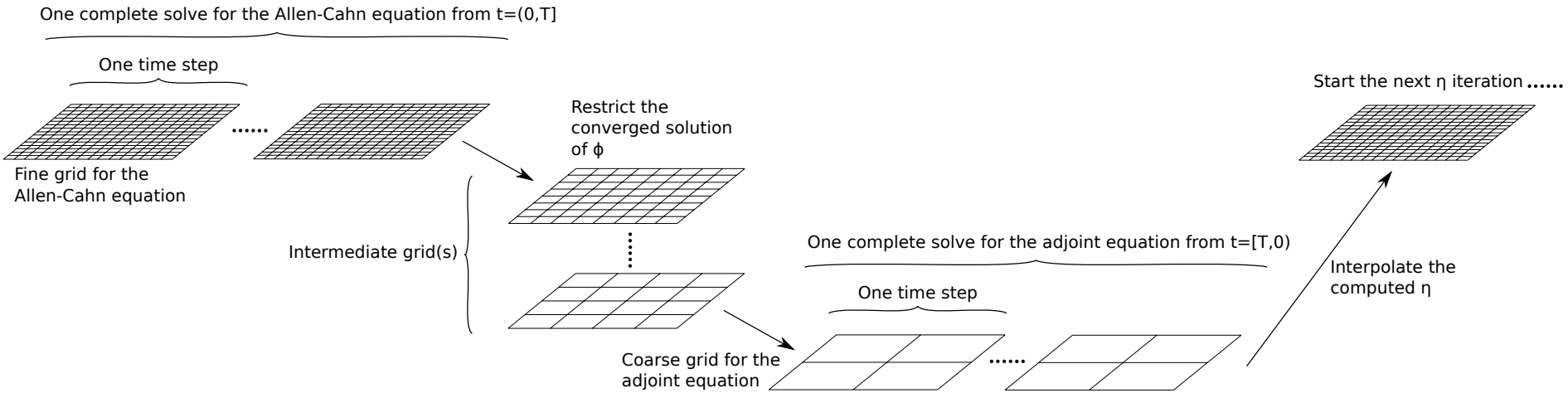
and we update the control as

$$\eta^{\ell+1} = \eta^{\ell} - \alpha \left(\theta \eta^{\ell} + \frac{1}{\epsilon} p^{\ell} \right).$$

Numerical challenges

- Number of time steps
- Memory requirement (let's consider double precision and 100 time steps)
  - 2-D: 512^2 requires 0.4 gigabytes
  - 3-D: 512^3 requires 215 gigabytes

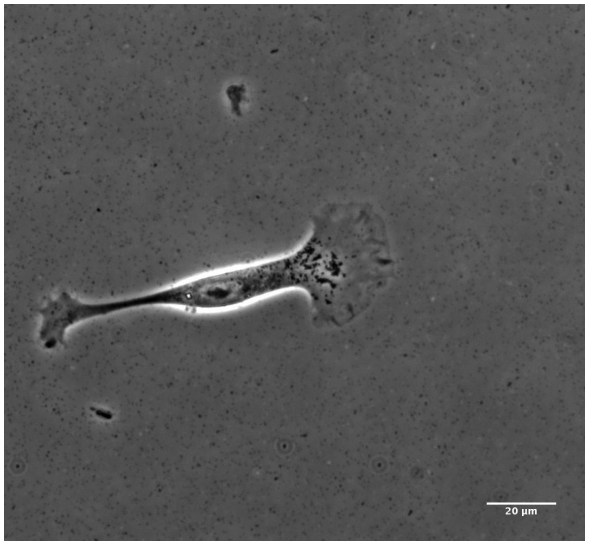
Two-grid solution strategy

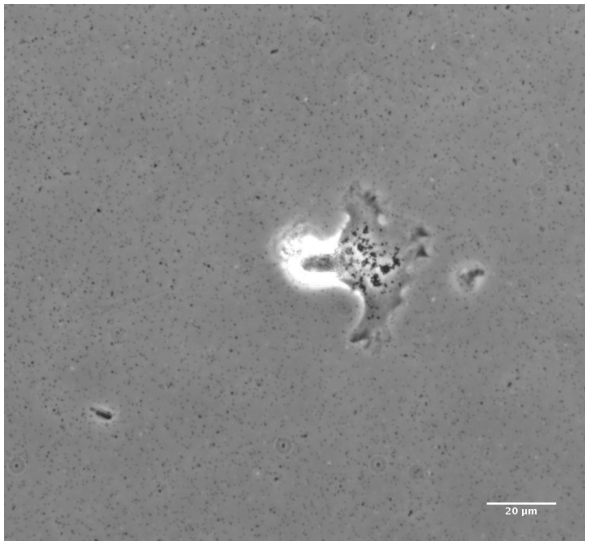


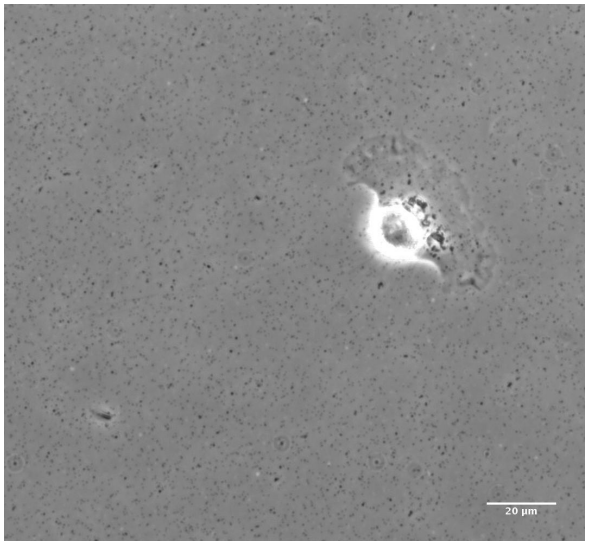
Real world example (1)

Real world example (1)

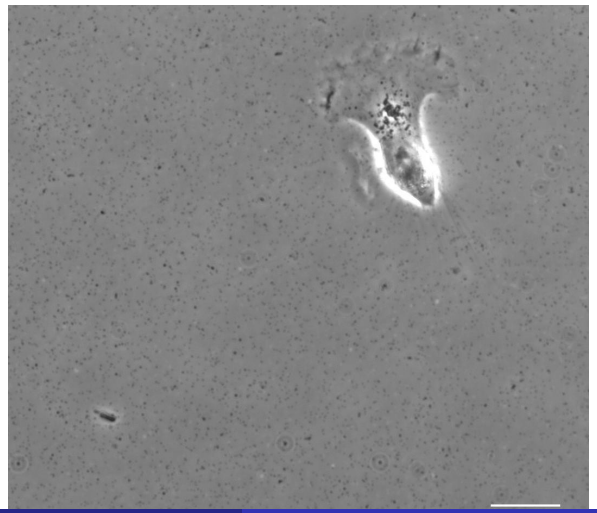
t=0







t=T



Real world example (1)

Real world example (2)

Euler number for topological changes

We compute this Euler number for these time steps with an "optimized" control η:

$$\mathcal{X} = \frac{1}{2\pi(a-b)} \int_{\Omega(a,b)} \left(-\triangle\phi + \frac{\nabla|\nabla\phi|^2 \cdot \nabla\phi}{2|\nabla\phi|^2} \right) dx.$$

Q. Du et al. *J. Appl. Math.*, 2005

Real world example (2)

A 3-D example